Introducción al desarrollo web (iDESWEB) - 3ª ed.

Práctica 7: JavaScript, Modelo de Objetos de Documento y cookies

1. Objetivos

- Aprender a manejar el DOM de una página web para manipular su contenido.
- Aprender a seleccionar un estilo alternativo mediante el DOM y JavaScript.
- Conocer el concepto de *cookie* y sus posibles usos.
- Aprender a utilizar las *cookies* con JavaScript.

2. Recursos

¿Qué son las cookies?

- **RFC 2965 HTTP State Management Mechanism**[1]: documento oficial que especifica el uso de las cookies para lograr una comunicación con HTTP que conserve el estado.
- **Cookie**[2]: definición en la Wikipedia de cookie, explica qué son, cómo se usan, algunos inconvenientes y alternativas a su uso.
- **Cookie Central**[3]: explica qué son las cookies y contiene ejemplos de uso.
- **Todo sobre la cookies**[4]: explica qué son las cookies y contiene ejemplos de uso en JavaScript.

¿Cómo se emplean las cookies con JavaScript?

- **JavaScript Cookies**[5]: cómo utilizar las cookies en JavaScript.
- **Las cookies**[6]: explicación en español del uso de las cookies en JavaScript.

3. ¿Qué tengo que hacer?

En la segunda práctica de CSS realizaste un estilo alternativo para tu sitio web. Algunos navegadores permiten a través de su menú seleccionar el estilo alternativo, mientras que otros navegadores como Microsoft Internet Explorer 6 o 7 no lo permiten[7]. Además, la mayoría de los usuarios desconocen que existe la posibilidad de cambiar el estilo de una página web (no saben que existe esa opción en el navegador web). Por eso, en esta práctica debes permitir que el usuario seleccione el estilo alternativo que desea desde la propia página web para que esta opción esté disponible en cualquier navegador y sea claramente visible para el usuario.

Por otro lado, cuando se selecciona un estilo alternativo, al pasar de una página web a otra de un mismo sitio web a través de un enlace no se conserva la selección del estilo alternativo y se muestra el estilo principal. Tienes que emplear las cookies para conservar el estilo seleccionado entre diferentes páginas e incluso entre diferentes visitas al sitio web.

[1] http://tools.ietf.org/html/rfc2965
[2] http://es.wikipedia.org/wiki/Cookie
[3] http://www.cookiecentral.com/c_concept.htm
[4] http://www.iec.csic.es/criptonomicon/cookies/default.html
[5] http://www.w3schools.com/js/js_cookies.asp
[6] http://www.mailxmail.com/curso/informatica/javascript/capitulo21.htm
[7] Por fin, Microsoft Internet Explorer 8 ya incorpora está característica, aunque no hace exactamente lo mismo que hacen otros navegadores.

4. ¿Cómo lo hago?

4.1. Selección de un estilo alternativo

A través del DOM podemos acceder al atributo `disabled` de la etiqueta `<link/>` que permite desactivar[8] (valor `true`) o activar (valor `false`) una hoja de estilo enlazada mediante la etiqueta `<link/>`.

Una primera propuesta de código JavaScript para realizar el cambio podría ser lo siguiente, donde la función `estilo()` recibe un parámetro que indica el estilo que se quiere activar (y, por tanto, todos los demás se tienen que desactivar):

```
<script type="text/javascript">
<!--
function estilo(titulo) {
  var arrayLink = document.getElementsByTagName('link');

  for(var i = 0; i < arrayLink.length; i++) {
    if(arrayLink[i].getAttribute('title') == titulo)
      arrayLink[i].disabled = false;
    else
      arrayLink[i].disabled = true;
  }
}
// -->
</script>
```

Sin embargo, este código puede fallar ya que la etiqueta `<link/>` tiene otros usos además de emplearse para enlazar una hoja de estilo. Además, ¿qué pasa con el estilo predeterminado que siempre se tiene que visualizar aunque se elija otro estilo alternativo? Tal como está escrito el código, el estilo predeterminado se desactivará la primera vez que se cambie de estilo alternativo.

La siguiente página es un ejemplo completo con una función de selección de estilo más compleja que contempla los problemas comentados anteriormente. En este ejemplo, un usuario puede seleccionar un estilo entre dos posibles estilos, además existe un estilo predeterminado y un estilo para sólo impresión:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es">
<head>
<title>Página con varios estilos</title>
<link rel="stylesheet" type="text/css" href="predeterminado.css" media="screen" />
<link rel="stylesheet" type="text/css" href="principal.css"  media="screen"
title="Principal" />
<link rel="alternate stylesheet" type="text/css" href="secundario.css"  media="screen"
title="Secundario" />
<link rel="stylesheet" type="text/css" href="impresion.css" media="print" />
<script type="text/javascript">
<!--
function estilo(titulo) {
  var arrayLink = document.getElementsByTagName('link');

  for(var i = 0; i < arrayLink.length; i++) {
    // Sólo aquellas etiquetas link que hacen referencia a un estilo
    // y que no sea para impresión
    if(arrayLink[i].getAttribute('rel') != null &&
       arrayLink[i].getAttribute('rel').indexOf('stylesheet') != -1 &&
       arrayLink[i].getAttribute('media') != 'print') {
      // Si tiene título es un estilo preferido o alternativo,
```

[8] Hay que tener cuidado con la propiedad `disabled` (desactivado) porque va al revés de lo que se podría pensar: para activar la etiqueta hay que poner su valor a `false`, y para desactivarla a `true`.

```
      // si no tiene título es un estilo
      // predeterminado y siempre tiene que utilizarse
      if(arrayLink[i].getAttribute('title') != null &&
         arrayLink[i].getAttribute('title').length > 0) {
        if(arrayLink[i].getAttribute('title') == titulo)
          arrayLink[i].disabled = false;
        else
          arrayLink[i].disabled = true;
      }
    }
  }
}
// -->
</script>
</head>
<body>
<ul>
<li><a href="javascript:estilo('Principal')">Principal</a></li>
<li><a href="javascript:estilo('Secundario')">Secundario</a></li>
</ul>
<p>
Una página web sencilla.
</p>
</body>
</html>
```

A continuación se muestra el código de la hoja de estilo predeterminada y de las dos hojas de estilo alternativas.

El código de la hoja de estilo predeterminada, almacenada en el fichero `predeterminado.css` es:

```
/* Estilo predeterminado */
html {
  font-size: 30px;
}

p {
  border: 5px solid #FF0000;
}
```

El código de la hoja de estilo principal, almacenada en el fichero `principal.css` es:

```
/* Estilo principal */
html {
  background-color: blue;
  color: white;
  font-family: Arial;
}

ul {
  list-style-type: none;
  background-color: white;
  color: red;
}

li {
  display: inline;
}

a {
```

```
  color: red;
}
```

El código de la hoja de estilo secundaria, almacenada en el fichero `secundario.css`:

```
/* Estilo secundario */
html {
  background: black;
  color: yellow;
}

ul {
  list-style-type: none;
}

li {
  display: inline;
}

a {
  color: green;
  font-weight: bolder;
}

p {
  font-size: 15px;
}
```

En la Figura 1 se muestra cómo se ve la página web con el estilo principal y en la Figura 2 se muestra con el estilo secundario.

4.2. Cookies

Una cookie es un fragmento de información que un navegador web almacena en el disco duro del visitante a una página web. La información se almacena a petición del servidor web, ya sea directamente desde la propia página web con JavaScript o desde el servidor web mediante las cabeceras HTTP, que pueden ser generadas desde un lenguaje de web scripting como PHP. La información almacenada en una cookie puede ser recuperada por el servidor web en posteriores visitas a la misma página web.

Las cookies resuelven un grave problema del protocolo HTTP: al ser un protocolo de comunicación "sin estado" (*stateless*), no es capaz de mantener información persistente entre diferentes peticiones. Gracias a las cookies se puede compartir información entre distintas páginas de un sitio web o incluso en la misma página web pero en diferentes instantes de tiempo.

En JavaScript, se puede acceder a las cookies a través de la propiedad cookie del objeto `document`. Esta propiedad permite acceder a todas las cookies de una página web, pero el acceso no es directo, ya que la propiedad `cookie` devuelve una única cadena que contiene todas las cookies de la página. Para acceder a una cookie concreta, es necesario analizar la cadena para localizar su valor.

La siguiente página de ejemplo contiene las funciones `setCookie()` y `getCookie()` que facilitan el acceso a las cookies en JavaScript. La función `setCookie()` recibe tres parámetros: el nombre de la cookie (`c_name`), el valor de la cookie (`value`) y la fecha de caducidad como el número de días a partir de la fecha actual (`expiredays`). La función `getCookie()` recibe sólo un parámetro, el nombre de la cookie que se quiere recuperar (`c_name`). En este ejemplo se emplea un cookie para almacenar la fecha de la última visita a la página web; la cookie tiene una validez de un año (365 días).

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es">
<head>
<title>Ejemplo de uso de cookies</title>
```

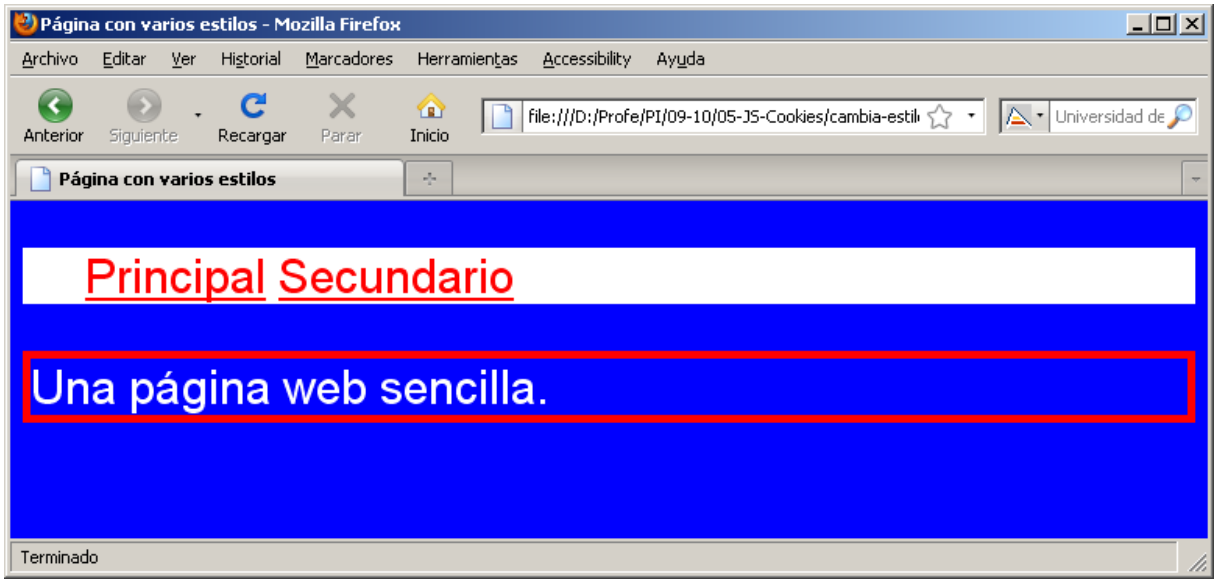

Figura 1: Estilo principal

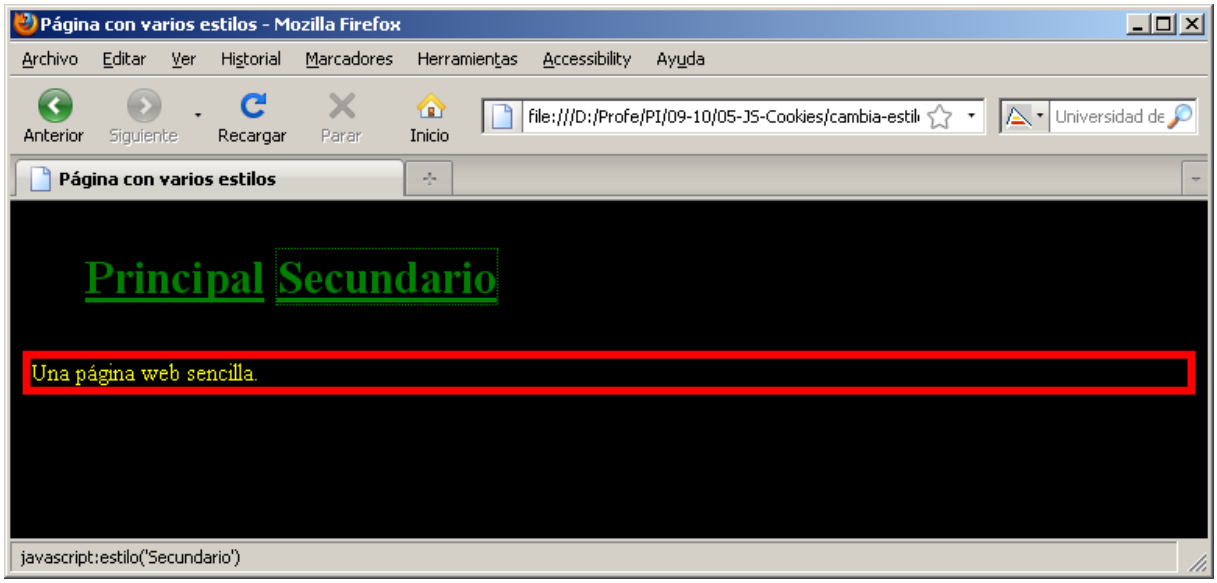

Figura 2: Estilo secundario

```
<script type="text/javascript">
<!--
// Código de http://www.w3schools.com/js/js_cookies.asp
function setCookie(c_name, value, expiredays) {
  var exdate = new Date();
  exdate.setDate(exdate.getDate() + expiredays);
  document.cookie = c_name + "=" + escape(value) +
    ((expiredays == null) ? "" : ";expires="+ exdate.toGMTString());
}

function getCookie(c_name) {
  if(document.cookie.length > 0) {
    c_start = document.cookie.indexOf(c_name + "=");
    if(c_start != -1) {
      c_start = c_start + c_name.length + 1;
      c_end = document.cookie.indexOf(";", c_start);
      if(c_end == -1)
        c_end = document.cookie.length;
      return unescape(document.cookie.substring(c_start, c_end));
    }
  }
  return "";
}
// -->
</script>
</head>
<body>
<p>
<script type="text/javascript">
<!--
var ultimavez = getCookie('ultimavez');

if(ultimavez != null && ultimavez != "") {
  document.writeln("Tu última visita fue: " + ultimavez);
  setCookie('ultimavez', Date(), 365);
}
else {
  document.writeln("Bienvenido por primera vez a este sitio web");
  setCookie('ultimavez', Date(), 365);
}
// -->
</script>
</p>
</body>
</html>
```

Para borrar una cookie, se tiene que asignar a la cookie una fecha de caducidad (`expires`) en el pasado, es decir, una fecha anterior a la actual.

5. Recomendaciones

La página web **HTML DOM Link Object**[9] explica las propiedades de la etiqueta `<link/>` que son accesibles a través del DOM.

En el ejemplo de este enunciado, la lista de estilos que puede seleccionar el usuario está escrita directamente en la página web. Si se añade un nuevo estilo alternativo, es necesario añadirlo a mano a la lista de selección. ¿Se puede automatizar el proceso, es decir, se puede crear de forma automática la lista de estilos que puede seleccionar el usuario?

[9] http://www.w3schools.com/jsref/dom_obj_link.asp

¡Cuidado! Por defecto, Google Chrome no proporciona soporte para el manejo de cookies a nivel local[10] (`file://`). Para realizar esta práctica, puedes activar el soporte de las cookies a nivel local con el modificador `--enable-file-cookies` al ejecutar Google Chrome o, mucho más sencillo, puedes usar otro navegador.

[10]Support cookies on file://: https://code.google.com/p/chromium/issues/detail?id=535